Generalized Kolmogrov Complexity and the Structure of Feasible Computations

Juris Hartmanis

TR 83-573
September 1973

Department of Computer Science
Cornell University
Ithaca, New York 14853

GENERALIZED KOLMOGOROV COMPLEXITY
AND THE STRUCTURE OF FEASIBLE COMPUTATIONS

(Preliminary Report)

Juris Hartmanis
Department of Computer Science
Cornell University
Ithaca, New York 14853

Abstract

In this paper we define a generalized, two-parameter, Kolmogorov complexity of finite strings which measures *how much* and *how fast* a string can be compressed and we show that this string complexity measure is an efficient tool for the study of computational complexity. The advantage of this approach is that it not only classifies strings as random or not random, but measures the amount of randomness detectable in a given time. This permits the study how computations change the amount of randomness of finite strings and thus establish a direct link between computational complexity and generalized Kolmogorov complexity of strings. This approach gives a new viewpoint for computational complexity theory, yields natural formulations of new problems and yields new results about the structure of feasible computations.

Introduction

Originally, Kolmogorov complexity of finite strings was introduced independently by A.N. Kolmogorov [Ko] and G. Chaitin [C1, C2] to give precise computational meaning to the concept of *randomness of a finite string*. Since then these concepts have been studied extensively and found many interesting applications in mathematics and computer science [C3, KK, PSS, Si].

The Kolmogorov complexity of a finite string is the length of the shortest program it can be computed from. Intuitively, it measures the amount *randomness* or *information* contained in the string. A string is *random* if it cannot be compressed. Unfortunately, this definition of randomness puts no limits on the time of computation of the given string from its compressed encoding and therefore makes randomness an undecidable property. As a matter of fact, the set of random strings is immune, in the sense that any recursively enumerable subset of the set of random strings must be finite. Stated differently, in any axiomatizable and sound mathematical system we can prove only a finite number of strings random and, more explicitly, no string longer than the description of the formal system can be proven random in the system [Ko, C3].

More recently time-bounded versions of Kolmogorov complexity have been studied and used to derive new results in computational complexity theory [Si, KK].

In this paper we define and study a generalized, two-parameter Kolmogorov complexity measure for finite string which measures how *far* and *how fast* a string can be compressed. We investigate the properties of this measure, reformulate some classic computational complexity problems and results in terms of these concepts and use them to derive new results about the structure of feasible computations.

Intuitively, the string x of length n, $|x|=n$, is in the Kolmogorov class

$$K[\log n, n^3]$$

if there exists a string of length $\log|x|$ or less, $|y|\leq \log|x|$, from which x can be computed in $|x|^3$ steps. The first parameter measures the amount of compression of the string and the second parameter gives the computational resource bound for the computation of the string in terms of the length of the computed string.

The advantage of the new approach is that it does *not only* classify strings as random or not (or polynomial-time random or not), but *measures the amount of randomness detectable in a given time.* This permits us to study, among other problems, how computation changes the amount of detectable randomness of individual strings and thus link computational complexity theory directly with the generalized Kolmogorov complexity of finite strings. For example, any honest function which maps low-polynomial-time random strings onto more random strings is not computable in polynomial time. Just as any honest function which maps less random strings (with unbounded computation time) onto more random strings is not recursive.

Another link to complexity theory is established by the very simple observations that by Cook's method [Co, GJ] we can show that every *higher nondeterministic time computation is equivalent to a higher deterministic computation followed by a polynomial time nondeterministic computation on the same problem with low Kolmogorov complexity.* This viewpoint gives a particularly clear interpretation of some known results in [Ha, HIS, HY] and permits the derivation of new results about the structure of feasible computations. For example, we show that the sparse set

$$SAT \cap K[\log n, n^2]$$

is a Cook-complete set for all other sparse sets in *NP* and derive related results for *PSPACE* and sets of other densities.

To further illustrate the utility of Kolmogorov complexity, we use it to give a new interpretation to oracle constructions and simplify several oracle constructions [BGS, HM, SB]. We also exhibit conditions in terms of Kolmogorov complexity under which there exist *NP* complete sets that are not polynomial-time isomorphic [BH]. At the present it is not known whether these conditions hold, but at least they exhibit *NP* complete sets which are not known to be p-isomorphic to *SAT*. For related work see [Ma, Yo].

Finally, we apply these concepts to make some more speculative observations about the nature of theorem proving and use of guessing in computations.

We believe that this study contributes not only some interesting new results and proof techniques but also a new viewpoint and concepts to the study of computational complexity. In particular, we believe that the study of how the Kolmogorov complexity of a problem statement is related to the Kolmogorov complexity of its solution has already yielded some initial insights and suggests many other interesting problems. A hope is that this may be a tool to explore and classify subclasses of *NP* complete problems which are easier to solve than the unrestricted problem.

Generalised Kolmogorov Complexity

The generalized Kolmogorov complexity measures ***how far*** a given string can be compressed and ***how easily*** it can be recomputed from the shortened representation.

For a universal Turing machine M_u and functions g and G mapping natural numbers into natural numbers, let

$$K_u[g(n),G(n)] = \{x \mid (\exists y)[\,|y| \le g(|x|) \text{ and } M_u(y) = x \text{ in } G(|x|) \text{ or fewer steps}]\}.$$

Thus the first function, g, measures by how much the string x must be compressed and G measures how much time is needed to compute x from y, where $|y|$ denotes the length of y. Note that the amount of time needed for the computation of x is expressed as a function of the length of x, namely $|x|$.

For example, $K[\log n, n^2]$ consists of all the strings x, $x \in \Sigma^*$, for which there exist strings y, $|y| \le \log |x|$ and $M_u(y)$ computes x in $|x|^2$ or fewer steps.

Though in this paper we are primarily interested in the above defined ***time-bounded*** Kolmogorov complexity, for the sake of completeness we define the corresponding ***space-bounded*** measure. For a universal Turing machine M_u, let

$$KS_u[g(n),G(n)]=\{x \mid (\exists y)[\,|y| \le g(|x|) \text{ and } M_u(y)=x \text{ using no more than } G(|x|) \text{ tape}\}.$$

Next we point out that there exist universal Turing machines which can be used in all complexity computations. We assume that we have fixed an input alphabet Σ, $|\Sigma| \ge 2$, to define K_u and KS_u.

Fact 1: There exists a universal Turing machine M_u such that for any other Turing machine M_v there exists a constant c such that

$$K_v[g(n),G(n)] \subseteq K_u[g(n)+c, cG(n)\log G(n)+c]$$

and

$$KS_v[g(n),G(n)] \subseteq KS_u[g(n)+c, cG(n)+c].$$

Proof: By the Hennie-Stearns simulation theorem [HS] of many-tape Turing machines on a two-tape Turing machine, we know that there exists a fixed universal machine M_u that can simulate any other machine, M_v, running in time $T(n)$ in time $dT(n)\log T(n)+d$, for some $d \ge 0$. Thus, picking $c \ge \max(d, |M_v|)$ we see that

$$K_v[g(n),G(n)] \subseteq K_u[g(n)+c, cG(n)\log G(n)+c].$$

The space bounded follows by a straight forward simulation argument. □

Because of the above result, which shows that the K_u and KS_u complexities are nearly optimal, we will drop the reference to a specific machine and write

$$K[g(n),G(n)] \text{ and } KS[g(n),G(n)],$$

respectively. It will be seen throughout this paper that all of our results are valid under the small changes of Kolmogorov complexity bounded in Fact 1.

It can also be shown that a slight increase in each parameter of Kolmogorov complexity increases the corresponding complexity class. We illustrate this with just one simple result, without giving the general separation results here.

Fact 2: $KS[\log n, n^2] \subset KS[\log n, n^2\log n]$,
$KS[\log n, n^2] \subset KS[2\log n, n^2]$.

Proof: We know, by a counting argument, that for all sufficiently large n

$$\Sigma^n - KS[\log n, n^2] \ne \emptyset.$$

Furthermore, for sufficiently large n we can simulate all n^2-tape computations of M_u on $n^2\log n$-tape and select the first string of length n not in $KS[\log n, n^2]$. Thus

$$KS[\log n, n^2] \subset KS[\log n, n^2\log n].$$

To prove the second containment relation we exploit the fact that for sufficiently large n we can pick a random string x which cannot be compressed and prefix it with the instruction for M_u to print an n-long string by repeating x, which can easily be done on n^2-tape. Since x cannot be computed from shorter strings we are guaranteed that

$$KS[\log n, n^2] \subset KS[2\log n, n^2]. \quad \square$$

Let $K[f(n)]$ denote the class of all strings x computable from strings of length less or equal to $f(|x|)$. Thus,

$$\Sigma^*-K[n-c],\ c>0,$$

is in essence the class of incompressible random strings as defined by Kolmogorov [Ko]. It is well known that for any computable, monotonic, unbounded function $f(n)$, the class $\Sigma^*-K[f(n)]$ is *immune*, i.e. it contains only finite recursively enumerable subsets.

Similarly, we have a related result for $\Sigma^*-KS[f(n),G(n)]$ and $\Sigma^*-K[f(n),G(n)]$. We state only one special case.

Fact 3: The set

$$\Sigma^*-KS[\log n, 2^n]$$

is 2^{cn}-space immune for any c, $c<1$; i.e. no infinite subset of $\Sigma^*-KS[\log n,\ 2^n]$ is in $SPACE[2^{cn}]$, $c<1$.

On the Structure of Feasible Computations

In this section we derive some structural results about P and NP by the use of Kolmogorov complexity and relate them to the higher deterministic and nondeterministic complexity classes.

There are two natural questions about problems which have simple Kolmogorov descriptions. We state them in terms of satisfiability of Boolean formulas:

1. Does low Kolmogorov complexity of problem descriptions imply that they are easy to solve? In particular, can we determine in polynomial-time for the simple Boolean formulas in $K[\log n, n^2]$ whether they are satisfiable, i.e. is

$$SAT \cap K[\log n, n^2] \in P?$$

2. How does the complexity of the problem statement affect the complexity of the solution? In particular, what can be said about the simplest satisfying assignments of formulas in

$$SAT \cap K[\log n, n^2]?$$

Recall that a set S, $S \cap \Sigma^*$, is sparse if there exists a k such that

$$|S \cap (\epsilon + \Sigma)^n| \leq n^k + k.$$

Theorem 4: $SAT \cap K[\log n, n^2] \in P$

$\Leftrightarrow$ No sparse sets in $NP-P$

$\Leftrightarrow NEXPTIME \stackrel{D}{=} \bigcup_{c \geq 1} NTIME[2^{cn}] =$

$\bigcup_{c \geq 1} TIME[2^{cn}] \stackrel{D}{=} EXPTIME.$

Proof: Follows by techniques developed in [Ha, HIS]. □

A more interesting fact is that

$$SAT \cap K[\log n, n^2]$$

is a ***Cook-complete set*** for all ***sparse sets*** in NP. We knew such sets existed [HY], but $SAT \cap K[\log n, n^2]$ can be viewed as ***natural*** complete set and therefore we knew that to deal with sparse sets in NP we only need information about "simple" formulas in SAT.

Note that, as shown later, there may exist sparse sets S in NP which are not subsets of $K[c \cdot \log n + c, n^k + k]$ for any $c, k \geq 1$, nevertheless, they are Turing reducible to $SAT \cap K[\log n, n^2]$. If a sparse set S in NP consists of simple strings, i.e. $S \subseteq K[c \cdot \log n, n^k]$, then S is many-one reducible to $SAT \cap K[\log n, n^2]$. We write $\leq_T^P$ and $\leq_M^P$, respectively, for polynomial-time Turing and many-one reducibility.

Theorem 5: a) If $S \in NP$ and S is sparse then $S \leq_T^P K[\log n, n^2] \cap SAT$.

b) If $S \in NP$ and $S \subseteq K[c \cdot \log n, n^k]$ then $S \leq_M^P K[\log n, n^2] \cap SAT$.

Proof: A refinement of the proof in [HY] that there exist sparse sets in NP that are complete for all other sparse sets in NP. □

The corresponding results for $PSPACE$ are considerably stronger, because we can easily enumerate a set of $PSPACE$ machines which accept all sparse sets and only sparse sets in $PSPACE$. It is not known whether a corresponding enumeration exists of NP machines which yield all sparse sets and only sparse sets of NP.

Let QBF denote the set of satisfiable quantified Boolean formulas, which is known to be $PSPACE$ complete. We say that a set S, $S \subseteq \Sigma^*$, has density $\delta(n)$ if

$$|S \cap (\epsilon + \Sigma)^n| \leq \delta(n).$$

The following result was obtained jointly with Yaacov Yesha.

Theorem 6: For any sparse set S in $PSPACE$,

$$S \leq_T^P K[\log n, n^2] \cap QBF$$

$$S \leq_M^P KS[\log n, n^2] \cap QBF.$$

Proof (outline): The first completeness result follows by the methods developed in [HY].

To prove the second completeness result, we observe that we can easily enumerate a sequence of machines M_{i_1}, M_{i_2}... which run on n-tape and accept no more than n strings of length n, and that every $PSPACE$ set of this type is accepted by one of these machines. From this sequence of machines we can construct a sparse universal set U in $PSPACE$,

$$U = \{1^{i_j} \# x \mid x \in L(M_{i_j})\}.$$

By a simple padding argument we see that every other sparse set in $PSPACE$ is $\leq_M^P$-reducible to U.

Since U is in $PSPACE$ and sparse, we know that for some $c > 0$,

$$U \subseteq KS[c \log n, n^c].$$

Since U is one-one-honestly reducible in polynomial-time to QBF and since such reductions cannot essentially increase the complexity of strings, we see that for some d,

$$U \leq_M^P QBF \cap KS[d \log n, n^d].$$

But then, by a padding argument,

$$U \leq_M^P QBF \cap KS[\log n, n^2]. \ \Box$$

The previous result can be extended to other densities.

Corollary 7: Let $\delta(n)$, $n \leq \delta(n) \leq 2^n$, be a monotonically increasing, $PSPACE$ computable density function. Then for any $\delta(n)$-dense set S in $PSPACE$

$$S \leq_T^P QBF \cap K[\lceil \log \delta(n) \rceil, n^2]$$

and

$$S \leq_M^P QBF \cap KS[\lceil \log \delta(n) \rceil, n^2].$$

It seems quite unlikely that all sparse sets in NP could be many-one reducible to $SAT \cap K[\log n, n^2]$. The next result shows that such an assumption has some interesting consequences.

Theorem 8: If $NEXPTIME = \Delta_2^E$ and every sparse set S in NP is many-one reducible to

$$SAT \cap K[\log n, n^2],$$

then $NEXPTIME = EXPTIME$ and therefore all sparse sets of NP are automatically in P.

Proof: The condition $NEXPTIME = \Delta_2^E$ is used to compute the minimal solutions of F_i in S and construct a new sparse set S' in NP which has the self-reducibility property (note that sparse sets of satisfiable formulas may not have self-reducibility). Then, by using a Berman-Mahaney like tree labeling argument [Ma] we show that reducibility of S' to a subset of $K[\log n, n^2]$ implies that S' is in P. □

The previous result can be used to show that there exist oracles for which all sparse sets in NP^A are not of low Kolmogorov complexity.

Corollary 9: There exists an oracle A such that for some sparse set $S \in NP^A$ S is not a subset of

$$K^A[c \log n, n^c],$$

for any $c > 0$.

Proof: There exists an oracle A (V. Sewelson) such that

$$EXPTIME^A \neq NEXPTIME^A = \Delta_2^{E(A)}.$$

Therefore, by Theorem 8 we conclude that not every sparse set in NP^A can be subsets of

$$K^A[c \log n, n^c],$$

for some c, or else $EXPTIME^A \neq NEXTIME^A$. □

Next we consider some relations between the complexity of problem descriptions and the complexity of their solutions.

Clearly, if all Boolean formulas F in $SAT \cap K[\log n, n^2]$ have some satisfying assignment in $K[c \log n, n^c]$, for a fixed c, then we can compute all possible strings in $K[c \log n, n^c]$ and determine in p-time if $F \in SAT \cap K[\log n, n^2]$. Therefore, we conclude:

Corollary 10: If every formula F in $K[\log n, n^2]$ has a satisfying assignment in $K[c \log n, n^c]$, for a fixed $c > 0$, then

$$SAT \cap K[\log n, n^2] \in P \text{ and } EXPTIME = NEXPTIME.$$

The following result establishes a necessary and sufficient condition linking the complexity of the Boolean formula to the complexity of its minimal solution.

Theorem 11: There exists a $c \geq 0$ such that the minimal solutions of formulas in $K[\log n, n^2] \cap SAT$ are in $K[c \log n + c, n^c + c]$ if and only if

$$\Delta_2^E = EXPTIME.$$

Proof: The proof exploits the fact that minimal solutions of F in SAT form a complete set in $P^{SAT} = \Delta_2^P$. The formulation is:

$$\{(F_i, j, d) \mid \text{the } j\text{th digit of minof } F_i \text{ is } d\}$$

is complete for Δ_2^P. Therefore, the minimal solutions of F in $SAT \cap K[\log n, n^2]$ by upward translation [HIS] form a complete set in Δ_2^E. Here the formulation is:

$\{(i, j, d) \mid$ the ith formula in $SAT \cap K[\log n, n^2]$ has d
as the jth digit in its min $\square$

Finally, we note that the assumption that

$$NEXPTIME \cap CoNEXPTIME \neq EXPTIME$$

implies that there exists a set S in NP,

$$S \subseteq SAT \cap K[\log n, n^2],$$

for which no polynomial time algorithm can compute a satisfying assignment for each F in S.

Theorem 12: $NEXPTIME \cap CoNEXPTIME \neq EXPTIME$ implies that there exists a set S in NP, $S \subseteq SAT \cap K[\log n, n^2]$ such that for all $c > 0$

$$\text{Min Sol}[S] \not\subseteq K[c \log n + c, n^c + c].$$

Furthermore, no set of solutions for all F_i in S is contained in $K[c \log n + c, n^c + c]$ for any $c > 0$.

Proof: The proof of this result is based on a technique developed by Borodin and Demers [BD] and the up-down method from [HIS]. $\square$

Non Isomorphic NP Complete Sets

In this section we illustrate another use of Kolmogorov complexity by exhibiting conditions under which there exist NP complete sets that are not p-isomorphic [BH].

Since the original work of Cook and Karp [Co, Ka] established the importance of NP complete problems, hundreds of different NP complete problems have been discovered in computer science and mathematics [AHU, GJ]. In spite of the different areas of origin, all the natural NP complete problems exhibit deep similarities. As a matter of fact, it has been shown that all the natural NP complete problems are polynomial-time isomorphic [BH]; thus in a very strong technical sense they all are similar. This led to the conjecture that all NP complete sets are p-isomorphic [BH].

It should be observed that if $P = NP$ then there exist non-isomorphic NP complete sets, thus a proof that all NP complete sets are p-isomorphic implies that $P \neq NP$. Still, even under the assumption that $P \neq NP$, there is no proof that all NP complete sets are

p-isomorphic. Recently, P. Young has exhibited new NP complete sets for which it is not known whether they are p-isomorphic to SAT, the satisfiable Boolean formulas in conjunctive normal form [Yo].

Using Kolmogorov complexity we give conditions under which there exist non-isomorphic NP complete sets. Informally stated, *if the satisfiability of Boolean formulas of low Kolmogorov complexity can be determined in polynomial time, then there exist NP complete sets that are not p-isomorphic to SAT.*

We explore other consequences of the above assumption and show that it forces certain higher deterministic and nondeterministic time classes to collapse. Intuitively, this has a nice interpretation: we know that $P=NP$ implies that there exist very dissimilar NP complete sets, i.e. finite and infinite. The partial collapse of the higher deterministic and nondeterministic time classes permits us to keep $P \neq NP$ but already allows NP complete sets which are not p-isomorphic.

It should also be noted that recently S. Kurtz [Ku] has constructed an oracle B such that $P^B \neq NP^B$ and there exist NP^B sets complete under $\leq_M^{P^B}$-reduction that are not p^B-isomorphic.

We say that a function $f:\Sigma^* \rightarrow \Gamma^*$ is *honest* if there exists a k such that

$$(\forall x)[\,|f(x)| \leq |x|^k + k \text{ and } |x| \leq |f(x)|^k + k],$$

i.e. f neither shrinks nor stretches x more than polynomially.

In our following results we will exploit the fact that an honest polynomial-time computable function cannot increase the Kolmogorov complexity of string by very much. Intuitively stated, a function which maps low Kolmogorov complexity strings onto high complexity strings cannot be an honest polynomial-time function.

Lemma 13: Let f be an honest function in P, then for all t, $t \geq 1$,

$$f(K[\log\log n, n^t]) \subseteq K[\log n, n^{\log n}] \text{ a.e.}$$

Proof: Since f is in P and honest, there exists a k and a v such that for all x

$$|x| \leq |f(x)|^k + k,\ |f(x)| \leq |x|^k + k$$

and $M_v(x)$ computes $f(x)$ in polynomial-time in $|x|$. For all x in $K[\log\log n, n^t]$ there exists a y, $|y| \leq \log\log |x|$, such that

$$M_u(y)=x \text{ in } |x|^t \text{ steps.}$$

Therefore, $f(x)=M_v(M_u(y))$ is computable in polynomial-time in $|f(x)|$, which implies that $f(x)$ is computable in polynomial-time in $|f(x)|$ by M_u from y', $|y'| \leq |y| + c_0$, for a fixed c_0.

Since for any $p \geq 1$ and sufficiently large n

$$\log\log(n^p + p) \leq \log n \text{ and } n^p \leq n^{\log n},$$

we see that

$$f(K[\log\log n, n^t]) \subseteq K[\log n, n^{\log n}] \text{ a.e. } \Box$$

We will now show that the assumption that the satisfiability of Boolean formulas of low Kolmogorov complexity can be determined in polynomial-time implies that there exist complete sets for NP which are not p-isomorphic. Clearly, we assume that $P \neq NP$ or there trivially exist complete NP sets which are not p-isomorphic.

Theorem 14: If there exists a set S_0 in P such that $S_0 \subset SAT$ and

$$K[\log n, n^{\log n}] \cap SAT \subseteq S_0,$$

then $SAT - S_0$ is an NP complete set which is not p-isomorphic to SAT.

Proof: To see that $SAT - S_0$ is $\leq_M^P$-complete for NP we note that

$$SAT \leq_M^P SAT - S_0$$

by reducing any F in S_0 to a fixed satisfiable Boolean formula not in S_0, recall S_0 is in P and $P \neq NP$, and all other strings are reduced to themselves.

At the same time, SAT and $SAT - S_0$ cannot be polynomial-time isomorphic since an isomorphism h is an honest function in P and therefore cannot map the simple strings in S_0, which are in SAT, onto the harder strings in $SAT - S_0$. More precisely, by Lemma 13,

$$h(SAT \cap K[\log\log n, n^3]) \subseteq K[\log n, n^{\log n}] \text{ a.e.}$$

and, since $SAT \cap K[\log\log n, n^3] \neq \emptyset$, we have

$$h(SAT \cap K[\log\log n, n^3]) \not\subseteq SAT - S_0.$$

Thus we see that

$$SAT \text{ is not } p\text{-isomorphic to } SAT - S_0. \ \Box$$

Intuitively, the assumption that

$$SAT \cap K[\log n, n^{\log n}] \subseteq S_0 \subset SAT$$

with S_0 is in P, asserts that if the Boolean formulas are sufficiently simple then their satisfiability is decidable in P. The above assumption can be weakened considerably:

Corollary 15: Let $g(n)$ be any unbounded, monotonically increasing function and $G(n)$ such that for all k

$$\lim_{n \to \infty} \frac{n^k}{G(n)} = 0.$$

Then

$$K[g(n), G(n)] \cap SAT \subseteq S_0 \subset SAT$$

and $S_0 \epsilon P$ implies that SAT is not p-isomorphic to $SAT - S_0$, which is an NP complete set.

Proof: A simple generalization of the previous proof. □

The above conditions which lead to non-isomorphic NP complete sets force some higher deterministic and nondeterministic complexity classes to collapse.

Corollary 16: If

$$K[\log n, n^{\log n}] \cap SAT \subseteq S_0 \subseteq SAT$$

and S_0 in P then

$$NEXPTIME = EXPTIME.$$

Proof: Since

$$K[\log n, n^3] \subseteq K[\log n, n^{\log n}] \ a.e.$$

We have

$$K[\log n, n^3] \cap SAT \subseteq S_0 \ a.e.,$$

but then

$$K[\log n, n^3] \cap SAT \epsilon P,$$

since S_0 is in P and so is $K[\log n, n^3]$. But $K[\log n, n^3] \cap SAT \epsilon P$, by methods developed in [HIS] can be shown to imply that

$$EXPTIME = NEXPTIME. \ \square$$

Similarly, the conditions of Corollary 15 imply that the correspondingly higher deterministic and nondeterministic time classes collapse.

It should be observed that if $P \neq NP$ then all the sets of the form

$$SAT - S,$$

with $S \subseteq SAT$ and S in P are NP complete. On the other hand, we cannot prove that these sets are p-isomorphic to SAT, since such proof would imply that for no $S_0 \in P$ do we have

$$SAT \cap K[\log n, n^{\log n}] \subseteq S_0 \subseteq SAT$$

and so far we have not been able to prove any separation results of this type.

Kolmogorov Complexity and Oracles

Relativization has played an important role in recursive function theory [Ro] as well as in computational complexity theory where it has been used to explore logical possibilities, limitations of proof techniques and assessing the difficulty of desired results [BGS, BSX, BLS, HIS, HM, Ku, LS, SB].

In this section we illustrate how the use of Kolmogorov concepts can simplify oracle constructions and bring new approaches to relativization. The main idea is *that a polynomial-time machine cannot compute from simple inputs complicated strings and therefore an oracle A constructed from complex strings is not accessible to P-machines working on simple strings while it is easily accessible to NP-machines.*

First we give a simple construction of an oracle A such that $P^A \neq NP^A$, which was first exhibited by Baker, Gill and Solovay [BGS].

Construction: By simple diagonalization we construct a set C, $C \subseteq \{1^{2^n} \mid n \geq 1\}$ such that

$$C \in TIME[n^{\log n}] - P.$$

Now place the first string of length 2^n from

$$K[\log n, n^{\log n}] - K[\log n, n^{\log\log n}]$$

in A iff $1^{2^n} \in C$.

Clearly,

$$P^A \neq NP^A,$$

since $C \in NP^A$ and C is not in P^A. To see the last assertion note that, for sufficiently large n, for input 1^{2^n} a p-machine cannot compute a string of length 2^n which could be in A, nor can it utilize shorter oracle strings, because they are already P-computable. Since $C \notin P$, we see that C is not in P^A.

By a similar construction we can give a very simple proof of a more recent result about immune sets in NP [BG, HM, SB]. A set C in NP is ***p-immune*** if C is infinite and contains no infinite subsets in P.

Fact 17: There exists a recursive oracle A such that NP^A has P^A-immune sets.

Construction: As in previous construction, obtain $C \subseteq \{1^{2^n} \mid n \geq 1\}$ by diagonalization to be a P-immune set in $TIME[n^{\log n}]$. The oracle A is constructed as above. Clearly C is infinite, in NP^A and P^A-immune, since again on the simple strings a P^A-machine cannot access the more complex oracle strings. □

It is interesting to note that the previous results are obtained with sparse oracles and that the P^A-machines are prevented from using the oracle because it consists of complex strings not computable in p-time from strings in 1^*. This contrasts strikingly with the following observation which shows the necessity for complex strings in the above oracles.

Fact 18: $P = NP$ if and only if for all oracles A, $A \subseteq K[\log n, n^2]$, $P^A = NP^A$.

For related results see [LS, BLS].

The above reasoning can be extended to random oracles with at most one element of each length.

Construction: For each n, $n \geq 1,2,\ldots$ toss a "fair coin" and if it comes up "heads" then toss the "fair coin" n times and place the resulting binary string of length n in A.

Fact 19: For the above oracle A,

$$P^A \neq NP^A$$

with probability 1.

Proof: To see this just observe that with probability 1 A is not computable and that with probability 1 there are random strings in A, not computable from 1^n. Therefore,

$$C = \{1^n \mid (\exists x)[x \in A, |x| = n]\} \in NP^A - P^A$$

with probability 1. □

Again it is interesting to note that for sparse random oracles $P^A \neq NP^A$ with probability 1, but that for some sparse oracles B such that $B \subseteq K[\log n, n^2]$, $P^B \neq NP^B$ if and only if $P \neq NP$ [BLS, BSX].

References

[AHU] Aho, A.V., Hopcroft, J.E., and Ullman, J.D. *The Design and Analysis of Computer Algorithms.* Addison-Wesley, Reading, Massachusetts, 1974.

[BD] Borodin, A., and Demers, A.J. Some comments on functional self-reducibility and the NP hierarchy. Tech. Rept. 76-284, Department of Computer Science, Cornell University, Ithaca, New York, July 1976.

[BG] Bennett, C.H., and Gill, J. Relative to a random oracle A, $P^A \neq NP^A \neq CoNP^A$ with probability 1. *SIAM J. Computing 10* (1981), 96-113.

[BGS] Baker, T., Gill, J., and Solovay, R. Relativization of the P =? NP question. *SIAM J. on Computing 4* (1975), 431-442.

[BH] Berman, L., and Hartmanis, J. On isomorphisms and density of *NP* and other complete sets. *SIAM J. Computing 6* (1977), 305-327.

[BLS] Book, R., Long, T., and Selman, A. Controlled relativizations of *P* and *NP*. *Lecture Notes in Computer Science 145* (1983), 85-90.

[BSX] Book, R., Selman, A., and Xu, M. Positive relativization of complexity classes. To appear in *SICOMP*.

[C1] Chaitin, G.J. On the length of programs for computing finite binary sequences. *J. ACM 13* (1966), 547-569.

[C2] Chaitin, G.J. A theory of program size formally identical to information theory. *J. ACM 22* (1975), 329-340.

[C3] Chaitin, G.J. Information – theoretic limitations of formal systems. *J. ACM 21* (1974), 403-424.

[Co] Cook, S.A. The complexity of theorem proving procedures. *Proc. 3rd Annual ACM Symp. Theory of Computing*, 1971, 151-158.

[GJ] Garey, M.R., and Johnson, D.S. *Computers and Intractability, A Guide to the Theory of NP-Completeness*. W.H. Freeman and Co., San Francisco, California, 1979.

[Ha] Hartmanis, J. On sparse sets in *NP–P*. *Information Processing Letters 16* (1983), 55-60.

[HIS] Hartmanis, J., Immerman, N., and Sewelson, V. Sparse sets in *NP–P*, *EXPTIME* versus *NEXPTIME*. *Proc. 15th ACM Symp. Theory of Computing*, 1983, 382-391.

[HoM] Homer, S., and Maass, W. Oracle dependent properties of lattices of *NP* sets. To appear in *Theoretical Computer Science*.

[HS] Hennie, F.C., and Stearns, R.E. Two-tape simulation of multi-tape Turing machines. *J. ACM 13* (1966), 533-546.

[HY] Hartmanis, J., and Yesha, Y. Computation times of *NP* sets of different densities. *Lecture Notes in Computer Science 154* (1983), Springer-Verlag, New York, 319-330.

[Ka] Karp, R.M. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, edited by R.E. Miller and J. Thatcher, Plenum Press, New York, 1972, 85-104.

[KK] Ko, K. Resource-bounded program-size complexity and pseudo-random sequences. To appear.

[Kol] Kolmogorov, A.N. Three approaches for defining the concept of information quantity. *Prob. Inform. Trans. 1,* (1965), 1-7.

[Ku] Kurtz, S.A. Private communication.

[LS] Long, T., and Selman, A.L. Relativizing complexity classes with sparse oracles. Unpublished abstract.

[Ma] Mahaney, S. Sparse complete sets for NP: solution of a conjecture of Berman and Hartmanis. *Proc. 21st Symp. Foundations of Computer Science,* 1980, 42-49.

[PSS] Paul, W., Seiferas, J., and Simon, J. An information-theoretic approach to time bounds for on-line-computation. *Proc. 12th ACM Symp. Theory of Computing,* 1980, 357-367.

[Ro] Rogers, H. *Theory of Recursive Functions and Effective Computability.* McGraw-Hill, New York, 1967.

[SB] Schoning, U., and Book, R.V. Immunity. *Automata Languages and Programming, Lecture Notes in Computer Science 154* (1983), Springer-Verlag, New York, 653-661.

[Si] Sipser, M. A complexity theoretic approach to randomness. *Proc. 15th ACM Symp. Theory of Computing,* 1983, 330-335.

[SS] Solovay, R., and Strassen, V. A fast Monte-Carlo test for primality. *SIAM J. on Computing 6* (1977), 84-85.

[Yo] Young, P. Some structural properties of polynomial reducibilities and sets in NP. *Proc. 15th ACM Symp. Theory of Computing,* 1983, 392-401.